High Performance Asynchronous Bit-Level Parallel Interface for Board-to-Board Inter Processor Communication

Faizal Arya Samman[1]
Dept. of Electrical Engineering
Faculty of Engineering, Universitas Hasanuddin
Kampus Gowa, Jl. Poros Malino Km. 20, Borongloe 92172
Email: faizalas@unhas.ac.id[1]

Thagiat Ahzan ADP[2] and Fandhi Nugraha[3]
Dept. of Electrical Engineering
Faculty of Eng., Universitas Hasanuddin
Email: ahzanadp@yahoo.co.id[2]
fendhinugraha@gmail.com[3]

***Abstract*—This paper presents a bit-level parallel communication interface used for inter processor communication separated on different printed circuit boards. A high performance board-to-board communication interface is important in modern supercomputers and portable computers or gadgets with multiple screen displays. We propose a recalibrated transmitter and receiver soft IP cores to support asynchronous handshake communication interface. The valid signal can be delayed for a few cycle to guarantee the metastability of data signals. The tuning of the delay can be recalibrated and tested during pre-implementation step. The flexibility to tune a correct valid delay time, which is set as minimum as possible as far as the data integrity can be guaranteed, enables the operation the communicating devices at its maximum performance. The proposed technique has been simulated using HDL-level simulation and has shown its expected performance with four testing scenarios.**

***Keywords*—Bit-parallel communication, Asynchronous communication interface, HDL Simulation, Transceiver, FPGA, IP core**

I. Introduction

Data communication is an important part of the nowadays information and communication technology industries. Most of devices are concerned with the transmission of data or information. Computers, gadget and other devices transmit data through a medium, wired or wireless. The need for high performance media access control protocol is a must in order to guaranteed the consumer satisfaction. The transmission bandwidth capacity or required between any two points in a point-to-point circuit depends on the average traffic conditions to be carried [1].

Data transmission between two or more devices, according to the data bit width, can be divided into two class, namely, bit serial and bit parallel communication. Serial communication is the process of transmitting and receiving data sequentially bit-by-bit. There are many commonly used serial communications, they are PS2, UART, I2C, SPI and USB, where all of them have been even standardized. PS2 is usually used as a communication interface between microcontroller with other devices such as mouse and keyboard. However, this communication protocol standard has been nearly obsolete, and replaced by USB. UART is commonly used to communicate data between two processors [2]. I2C and SPI commonly used for communication on inter-chip or inter-chip low-medium speed data-stream transfers such as ADC with microcontroller and sensor with microcontroller [3] [4], [5]. USB is the most common communication interface today. we can find it on port computer, printer port, mouse, keyboard and others.

In bit-level parallel communication, data transmission can be accomplished as much as N-bits through an N-bits data path. Thus, data transmission becomes faster, but on the other hand, the logic gate area becomes higher. The bit-level parallel communication interface is required for the transmission of large amounts of data and in a short time. To implement this, a standard module is required that ensures stability, reliability, and is able to work effectively in improving the work efficiency.

The development of board-to-board communication has its own challenges, namely metastability, crosstalk, and cross domain clock. Metastability is concerned with problem called data signals stability on a data path, which cannot be read before all data signals haven been in a steady-state condition. Accessing the data before the metastability is guaranteed can cause the lose of data integrity and validity. To solve this problem we can use an open loop or closed loop method with synchronizer. Meanwhile, we use a valid signalling method to ensure that the data have been steadily loaded before the recipient node read them from the physical link.

Data processing capability of a system depends not only on data processing devices but also influenced by data communication interface. The communication interfaces are connected directly to the physical links of between two communicating devices on each separated board. The bottleneck performance presented in the physical link will lower the system performance, regardless the higher working frequency speed of devices or processing elements. Hence, board-to-board data communication is important aspect in a high performance computing system, as it is also discussed in this paper. Supercomputers or high performance computers implemented on a huge number of rack boxes, and smart gadgets implemented with multiple display screens in different boards are good examples of the board-to-board inter processor data communication.

978-1-5386-2708-2/17/$31.00©2017/IEEE

II. Related Works and The Key Feature

Board-to-board data communication is usually classified into asynchronous data communication scheme, because naturally both communicating devices on the boards have different clock sources. The clock frequency and clock phase of the devices can accordingly be different.

The main problem in the asynchronous communication is as mentioned before the metastability. There are many techniques that can be utilized to overcome that problem. A transceiver is usually interfaced with a FIFO buffer with its back-end processing element. The FIFO buffer is implemented both in its transmitting side and in its receiving side. Therefore, we can design and implement a dual-clock FIFO buffer, which can be clocked with two different clock frequencies [6].

To overcome the metastability problem, we can also implement a source synchronous interface [7], where the sending node sends its clock signal through a clock path to the receiving node. The sent clock signal is certainly used to synchronized the transmitted data.

Another work by Jenning et al [8] presented also a transceiver for a board-to-board communication interface. In particular, the proposed technique replaced the wired data link with the deployment of wireless link, where the used carrier frequency is above 100 GHz. The wireless communication is efficient in term of the simplicity of the complex cabling. However, the wireless communications can be made only with single bit serial data communication, which has lower data rate compared to its counterpart bit-parallel data communication. However, we can implement wireless bit-parallel data communication with a specific reliable and robust communication protocol such, as multiple carrier frequencies or carrier codes.

This paper propose also another technique to overcome the metastability problem. The proposed technique is derived from our previous work [9]. However, we have made better improvements in term of reconfigurability of the valid delay settlement as the key feature of the interface. The time delay of the steady state conditions of the N-bit data signals on the N-bit data paths can be different depending on the length and the characteristic of the physical links, including the pattern of the metal wire paths on the board. Thus, during pre-implementation testing, we can configure and calibrate the correct valid delay time as minimum as possible as far as the data integrity can be guaranteed. Therefore, we can operate the communicating devices at its maximum performance. Our technique has been simulated using HDL-level simulation and has shown its performance with four testing scenarios.

III. Overviews of the Protocol and On-Chip Architecture

Fig. 1 presents the on-chip architecture of our communication interface. The figure present two interfaces, where each of them implemented is on a Field Programmable Gate Array (FPGA) device. Each interface consists of four components, i.e. two first-in first-out (FIFO) buffer, a transmitter core (TX) and a receiver core (RX). Each interface on the different FPGA mounted on different printed circuit board (PCB) has its own and different clock source. Eventually, they can also work with different frequency clocks. Thus, not only frequency but also the phase of the clocks of both FPGAs are probably not equal.

The data communication interface is full-duplex. Each interface is faced the other one through an N-bit data link and a valid-bit and an acknowledge (ack) signal link. The valid signal flows via the single bit valid path, and can be delayed for several cycles. The acknowledge signal flows back through the single-bit acknowledge path.

At sender side, a FIFO buffer is used to interface the TX module with a processor system bus which connecting also a memory module. Another FIFO is also used to interface the RX module with another processor system bus which connecting also a memory module. The sender processor produces data, and move them into the FIFO buffer. The TX module will receive the data and send them to the physical link. For a few cycle the valid signal is delayed and is set to inform the RX module at the other board side. The RX module will then send an acknowledge signal or flag soon after it receives the data and send them into the FIFO buffer.

IV. Simulation Result

The simulation results presented in this paper are categorized into 4 parts, The first category is the simulation where the transmitting and receiving nodes have the same working clock frequency, as presented in Section IV-A. The second category is the simulation where the transmitting and receiving nodes have the same working clock frequency but having different clock phase, as presented in Section IV-B. The third category is the simulation where the transmitting node has slower clock frequency than the receiving node, as presented in Section IV-C. The fourth category is the simulation where the transmitting node has faster clock frequency than the receiving node, as presented in Section IV-D.

In all simulation cases, we measured the number of clock cycles required by each data to be in the transmitter input terminal and the receiver output terminal for each different setting time of valid flag signal measured in number of clock cycles. Hence, we will see in the simulation results the data sequence number and the number of clock cycles.

A. Simulation with The Same Clock Frequency

Fig. 2 presents the simulation result of the required cycle-time of each data to be in transmitter side when the transmitter and the receiver's have same clock frequency. The figure shows that the required cycle-time is increased as the flag of the valid signal is delayed with more cycle times.

The communication performance or data rate at an instant cycle can be formally modelled as follows:

$$R = \frac{D}{T_C} \tag{1}$$

If D is the number of transmitted data words over the total cycle time T_C. The unit of R can be determined as the number of data words per cycle period.

If t_k is the cycle time to detect a transmitted data word k^{th} and t_{k-1} is the cycle time to detect the previous transmitted data word $(k-1)^{th}$, then formally the time-dependent data rate $R(t_k)$ at instant time t_k can be expressed as follows.

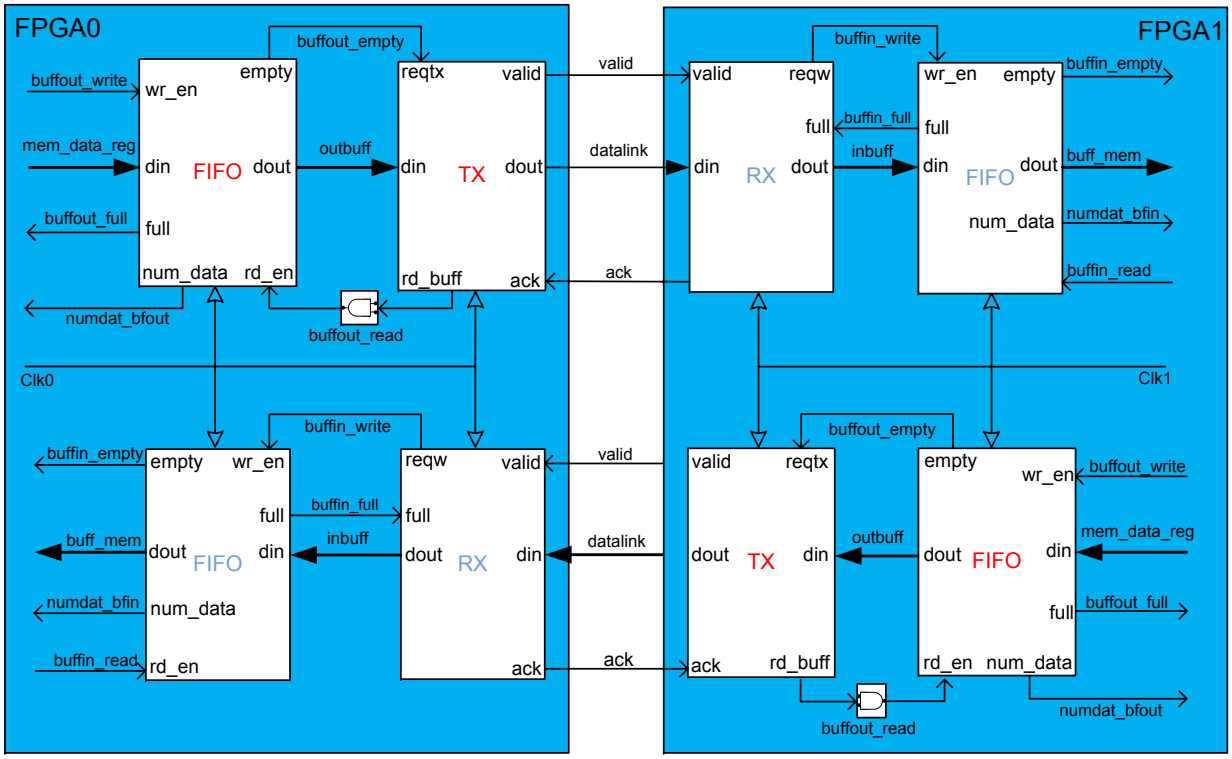

Fig. 1. Block Diagram of the transmitter module (TX) and the receiver module (RX).

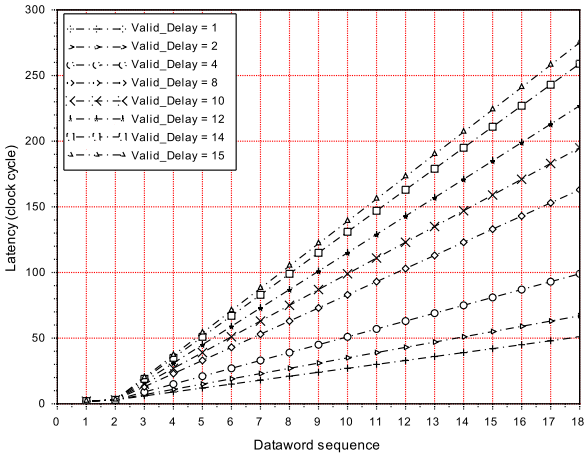

Fig. 2. Simulation Result of the required cycle-time of Data to be in Transmitter side when The Transmitter and The Receiver's have same clock frequency.

$$R(t_k) = \frac{1}{t_k - t_{k-1}} \tag{2}$$

Fig. 3 presents the simulation result of the required cycle-time of each data to be in receiver side when the transmitter and the receiver's have same clock frequency. The figure shows also that the required cycle-time is increased as the flag of the valid signal is delayed with more cycle times.

Based on the simulation result presented in Fig. 3 and the formal model shown in Eq. 2, the time to detect the 15^{th} and the 16^{th} data words for 2 cycles valid delay are respectively 62 and 66 cycle times. With 4 cycle time difference, then the data rate at the 66^{th} instant cycle is $\frac{1}{4} = 0.25$ data word per cycle. When the transmitter and receiver node works with 100 Mhz clock frequency or 0.1 ns clock period and the data width is 32-bit, then we can estimate that the data rate communication for that condition is $\frac{0.25 \times 32}{0.1\ ns} = 80$ Gbps. By using 64-bit data word width, the data rate can approach 160 Gbps.

For 14 cycles valid delay as presented in Fig. 3, the time to detect the 15^{th} and the 16^{th} data words are respectively 242 and 258 cycle times. With 16 cycle time difference, then the data rate at the 66^{th} instant cycle is $\frac{1}{16} = 0.0625$ data word per cycle. When the transmitter and receiver node works with 100 Mhz clock frequency or 0.1 ns clock period and the data width is 32-bit, then we can estimate that the data rate communication for that condition is $\frac{0.0625 \times 32}{0.1\ ns} = 20$ Gbps. By using 64-bit data word width, the data rate can approach 40 Gbps. Thus, by using CMOS technology with smaller minimum transistor feature/gate size in order to achieve a device with higher clock frequency, then a high performance communication interface can be realized.

From Fig. 2 and Fig. 3, we can see that the time to move a data from transmitter and receiver depends heavily not only on the valid setting delay but also affected by the delay time due to valid and ack delivery. The slope of each curve in the figures represents the data rate of the communication.

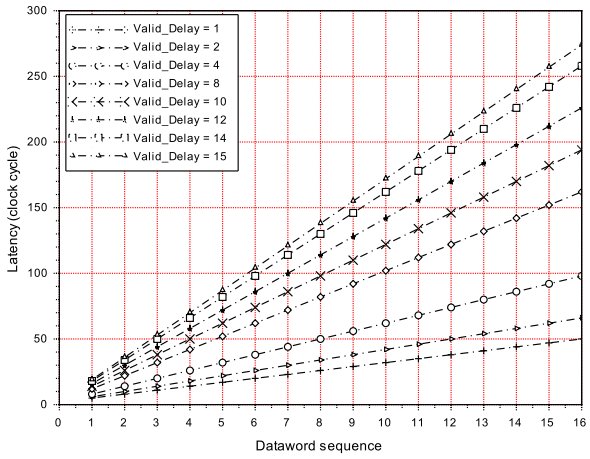

Fig. 3. Simulation Result of the required cycle-time of Data to be in Receiver side when The Transmitter and The Receiver's have same clock frequency.

B. Simulation with Different Clock Phase

Fig. 4 presents the simulation result of the required cycle-time of each data to be in transmitter side when the transmitter and the receiver's have same clock frequency, but both have different clock phase. The figure shows that the required cycle-time is increased as the flag of the valid signal is delayed with more cycle times.

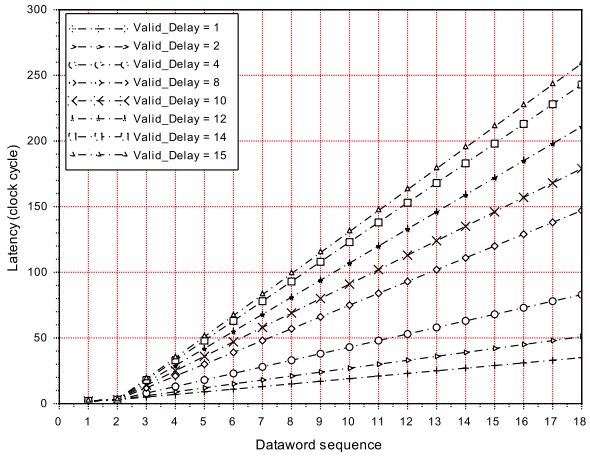

Fig. 4. Simulation Result of the required cycle-time of Data to be in Transmitter side when The Transmitter and The Receiver have same clock frequency, but have different clock phase.

Fig. 5 presents the simulation result of the required cycle-time of each data to be in receiver side when the transmitter and the receiver's have same clock frequency. The figure shows also that the required cycle-time is increased as the flag of the valid signal is delayed with more cycle times.

When we compare the results from Fig. 4 and Fig. 5 with the previous results from Fig. 2 and Fig. 3, then we can see that the time delay for the data transmission is not significantly affected by the phase difference of the clock. The significant effect occurs when the phase difference is 0.85π until 0.95π radian, and the delay time can increase by one clock cycle.

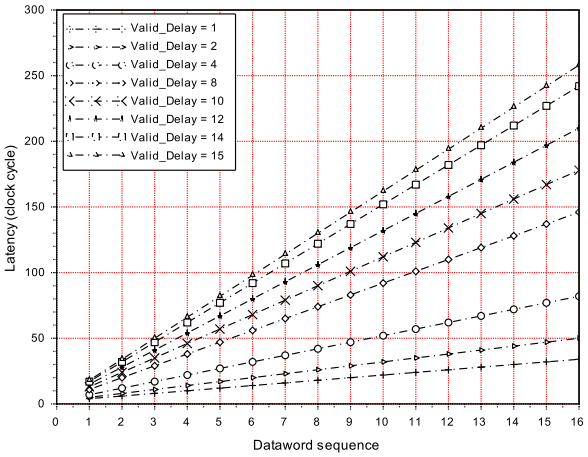

Fig. 5. Simulation Result of the required cycle-time of Data to be in Receiver side when The Transmitter and The Receiver have same clock frequency, but have different clock phase.

C. Simulation with Transmitter's Slower Clock Frequency

Fig. 6 presents a simulation result of the required cycle-time of data to be in transmitter input terminal when the transmitter's clock frequency is slower than the receiver's clock frequency. Figure . 7 presents a simulation result of the required cycle-time of data to be in receiver output terminal when the transmitter's clock frequency is slower than the receiver's clock frequency. Both figures show also that the required cycle-time is increased as the flag of the valid signal is delayed with more cycle times.

D. Simulation with Transmitter's Faster Clock Frequency

Fig. 8 presents a simulation result of the required cycle-time of data to be in transmitter input terminal when the transmitter's clock frequency is faster than the receiver's clock frequency. Fig. 9 presents a simulation result of the required cycle-time of data to be in receiver output terminal when the transmitter's clock frequency is faster than the receiver's clock frequency. Both figures show also that the required cycle-time is increased as the flag of the valid signal is delayed with more cycle times.

If we compare the simulation results of Fig. 6 and Fig. 8, we can see that the latency of the data with the slower transmitter clock frequency is higher than one with the higher transmitter clock frequency.

V. SYNTHESIS RESULT

The logic synthesis of parallel communication based on an FPGA is presented in this paper, because FPGAs are easy to configure and accordingly have low prototyping cost. The FPGA (Field Programmable Gate Array) has a characteristic of

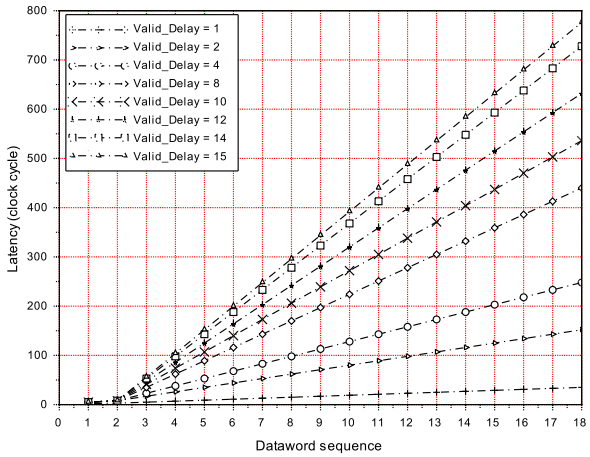

Fig. 6. Simulation Result of the required cycle-time of Data to be in Transmitter side when The Transmitter's clock is slower than The Receiver's clock frequency.

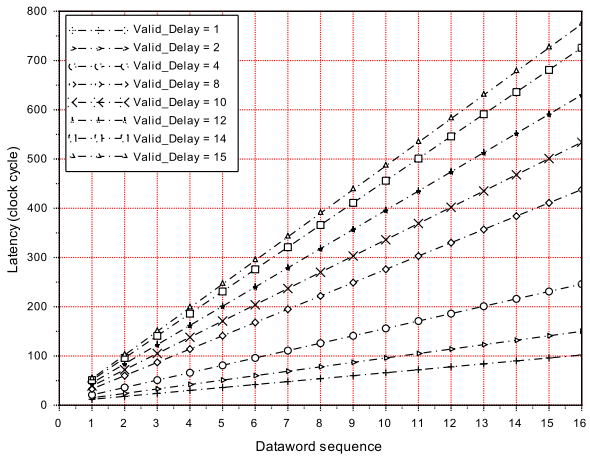

Fig. 7. Simulation Result of the required cycle-time of Data to be in Receiver side when The Transmitter's clock is slower than The Receiver's clock frequency.

the reconstruction, the rapidity, design flexibility and the high-density of logical resources [10] and can meet the increasingly complex logic demands [11].

We have synthesized our transmitter and receiver IP (Intellectual Property) cores using Cyclone III device with device number EP3C16F484C6, an FPGA device from Altera. By using 16-bit parallel data interface, the transmitter core requires about 31 logic elements and the receiver core requires 38 logic elements. From the synthesis data, we can seen that our cores have relatively low logic area. Unfortunately, we cannot compare this synthesis data with other bit-parallel communication interface techniques due to the lack of the FPGA-based synthesis data provided by the other parallel interfaces.

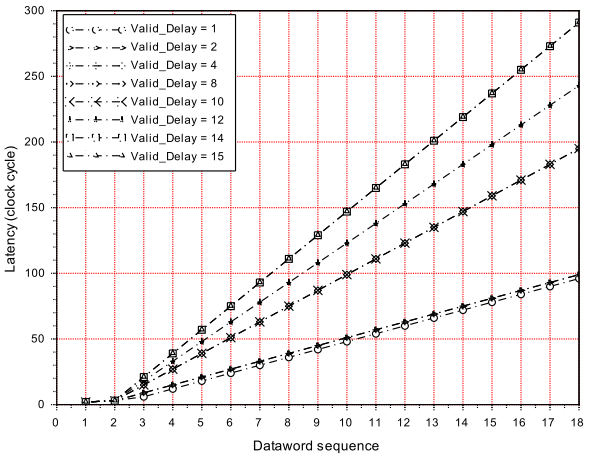

Fig. 8. Simulation Result of the required cycle-time of Data to be in Transmitter side when The Transmitter's Clock is faster than The Receiver's Clock frequency.

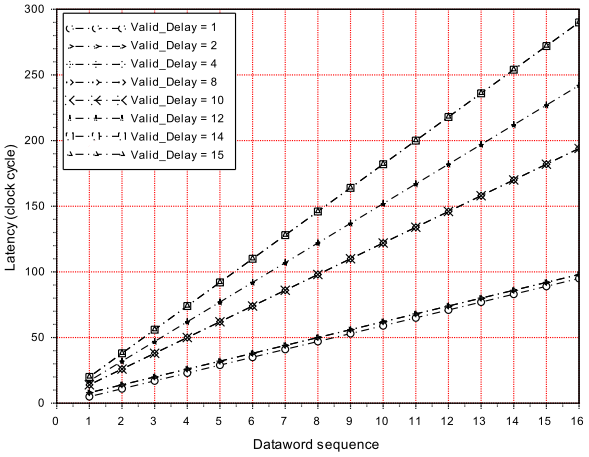

Fig. 9. Simulation Result of the required cycle-time of Data to be in the Receiver side when The Transmitter's Clock is faster than The Receiver's clock frequency

VI. Conclusions and Outlooks

This paper has presented reconfigurable transmitter and receiver IP cores with bit-level parallel interface used in board-to-board inter processor communications. The proposed transceiver IP cores allows us to implement bit-level parallel communications through physical links between devices mounted on different boards with the following operating clock conditions.

1) The sender and the receiver device have the same working clock frequency and clock phase.
2) The sender and the receiver device have the same clock frequency but have different clock phase.
3) The sender's clock frequency is lower than the receiver's clock frequency.
4) The sender's clock frequency is higher than the

receiver's clock frequency.

Under above working frequency conditions the transmitted data by the sender can be received well by the receiver node. In our HDL-level simulations, we use FIFO buffers with 16-slot data buffer. Therefore the data measurements, particularly on the receiver side, are made for only until the sixteenth datum. In the future, we will analyse the impacts of the FIFO buffer depth on the data communication performance, as well as the performance measurement for higher testing data volumes.

We have also made the performance estimation of the proposed communication interface. For 14 cycles valid delay for example as shown in the simulation result with the same clock frequency, the data rate at the last measured cycle time is 0.0625 data word per cycle. When the transmitter and receiver node works with 100 Mhz clock frequency or 0.1 ns clock period and the data width is 32-bit, then we can estimate that the data rate communication for that condition is $\frac{0.0625 \times 32}{0.1\ ns} =$ 20 Gbps. By using 64-bit data word width, the data rate can approach 40 Gbps. Therefore, we can potentially achieve a high performance communication interface, by using CMOS technology with smaller minimum transistor feature/gate size.

We have not implemented our design onto a CMOS standard-cell technology yet. However, the soft IP cores of the transmitter and receiver have been synthesized using a Cyclone III FPGA device from Altera. The total number of logic elements used on the FPGA device for both transmitter and receiver is about 69 logic elements, i.e. 31 for the transmitter core and 38 for the receiver core. This number is quite small and will potentially consume low logic area when we implement it using CMOS standard-cells in the future.

ACKNOWLEDGMENT

We gratefully acknowledge the Ministry for Research, Technology and Higher Education of the Republic of Indonesia for funding and supporting our research work entitled "Multi-processor Systems-on-Chip for Innovative Smart Gadgets with Multiple Touch Screen" under the scheme of "National Strategic Outstanding Research Grant" (Hibah Penelitian Unggulan Strategis Nasional or PUSNAS) with Grant Contract Number 005/SP2H/LT/DPRM/IV/2017 in the year 2017.

REFERENCES

[1] P. T. Kirstein, "Data communication by packet switching," *Electronics and Power*, vol. 19, no. 20, pp. 503–508, November 1973.

[2] U. Nanda and S. K. Pattnaik, "Universal asynchronous receiver and transmitter (uart)," in *2016 3rd International Conference on Advanced Computing and Communication Systems (ICACCS)*, vol. 01, Jan 2016, pp. 1–5.

[3] N. Anand, G. Joseph, S. S. Oommen, and R. Dhanabal, "Design and implementation of a high speed serial peripheral interface," in *2014 International Conference on Advances in Electrical Engineering (ICAEE)*, Jan 2014, pp. 1–3.

[4] M. Pulkkinen, L. Aaltonen, and K. Halonen, "Spi interface, mux-based synchronizer and dsp unit for a mems-based accelerometer," in *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2015, pp. 453–456.

[5] W. Andrysiewicz, D. Kocielnik, and M. Mikowicz, "I2c hardware master serial interface for asynchronous adcs," in *2015 IEEE International Symposium on Systems Engineering (ISSE)*, Sept 2015, pp. 77–81.

[6] R. W. Apperson, Z. Yu, M. J. Meeuwsen, T. Mohsenin, and B. M. Baas, "A scalable dual-clock fifo for data transfers between arbitrary and haltable clock domains," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 15, no. 10, pp. 1125–1134, Oct 2007.

[7] T. O. Dickson, Y. Liu, A. Agrawal, J. F. Bulzacchelli, H. A. Ainspan, Z. Toprak-Deniz, B. D. Parker, M. P. Beakes, M. Meghelli, and D. J. Friedman, "A 1.8 pj/bit 16 16 gb/s source-synchronous parallel interface in 32 nm soi cmos with receiver redundancy for link recalibration," *IEEE Journal of Solid State Circuits*, 2016.

[8] M. Jenning, B. Klein, R. Hahnel, D. Plettemeier, D. Fritsche, G. Tretter, C. Carta, F. Ellinger, T. Nardmann, M. Schroter, K. Nieweglowski, K. Bock, J. Israel, A. Fischer, N. U. Hassan, L. Landau, M. Dorpinghaus, and G. Fettweis, "Energy-efficient transceivers for ultra-highspeed computer board-to-board communication," in *2015 IEEE International Conference on Ubiquitous Wireless Broadband (ICUWB)*, Oct 2015, pp. 1–5.

[9] F. A. Samman, F. Philipp, and M. Glesner, "Reconfigurable interconnect infrastructure for multi-fpga-based adaptive multiprocessing systems," in *2011 1st International Workshop on Computing in Heterogeneous, Autonomous and Goal-Oriented Environments (CHANGE)*, March 2011, pp. 1–8.

[10] H. Huang, Z. Yan, and X. Zhang, "Dangerous rock testing system based on fpga," in *2016 First IEEE International Conference on Computer Communication and the Internet (ICCCI)*, Oct 2016, pp. 358–361.

[11] H. l. Peng and Y. l. Nie, "Design of serial communication interface based on fpga," in *2011 IEEE International Conference on Computer Science and Automation Engineering*, vol. 4, June 2011, pp. 410–414.